A Clustering and Traffic-Redistribution Scheme for High-Performance IPsec VPNs*

Pan-Lung Tsai, Chun-Ying Huang, Yun-Yin Huang,
Chia-Chang Hsu, and Chin-Laung Lei

Department of Electrical Engineering,
National Taiwan University, Taipei 106, Taiwan
{charles, huangant, yunyin, oberon}@fractal.ee.ntu.edu.tw
lei@cc.ee.ntu.edu.tw

Abstract. CPE-based IPsec VPNs have been widely used to provide secure private communication across the Internet. As the bandwidth of WAN links keeps growing, the bottleneck in a typical deployment of CPE-based IPsec VPNs has moved from the last-mile connections to the customer-edge security gateways. In this paper, we propose a clustering scheme to scale the throughput as required by CPE-based IPsec VPNs. The proposed scheme groups multiple security gateways into a cluster using a transparent self-dispatching technique and allows as many gateways to be added as necessary until the resulting throughput is again limited by the bandwidth of the last-mile connections. It also includes a flow-migration mechanism to keep the load of the gateways balanced. The results of the performance evaluation confirm that the clustering technique and the traffic-redistribution mechanism together create a transparent, adaptive, and highly scalable solution for building high-performance IPsec VPNs.

1 Introduction

As the Internet becomes the most widely used transport for information delivery, it is logical for people to move away from traditional private networks consisting of dedicated circuits and turn to VPNs, or leveraging the Internet for private communication, for the reason of cost savings and convenience [1]. In order to maintain data security during the transmission of private information over the public Internet, the technologies used to implement VPNs usually make use of various encryption and authentication techniques based on cryptographic operations in such a way that the so-called tunnels are established before data transmission and the data flowing through the tunnels is protected. A commonly referred to example is the tunnel mode defined in the IPsec standard [2].

Among all approaches, Customer Premise Equipment-based (CPE-based) VPNs are one of the most popular. Fig. 1 shows a VPN between two organizations and the location of the tunnels. With CPE-based VPNs, private communication among coalition members can be achieved without requiring Internet

* This work was supported in part by the Taiwan Information Security Center, National Science Council under the grant NSC 94-3114-P-001-001-Y.

D.A. Bader et al. (Eds.): HiPC 2005, LNCS 3769, pp. 432–443, 2005.

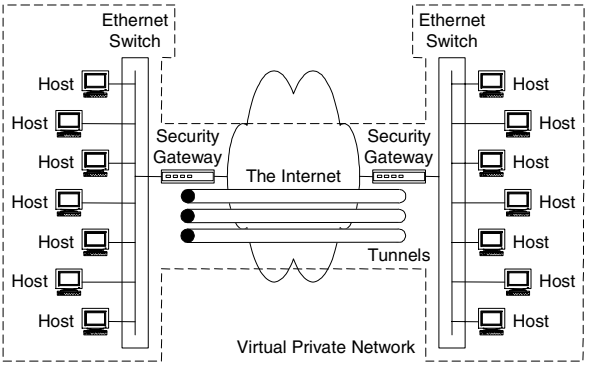

Fig. 1. Tunnels in a typical CPE-based VPN terminate on the respective security gateways residing in the both sides of the VPN

backbone (e.g., service providers) to offer special services other than network connectivity. Besides, as the IPsec protocol becomes a globally recognized standard in the industry, almost all CPE-based VPNs in the modern times run the IPsec protocol [3] and are potentially interoperable.

Despite the advantages such as interoperability and transparency (i.e., hosts behind the gateways are not required to implement the IPsec protocol), the centralized processing model limits the scalability of CPE-based IPsec VPNs. Since each security gateway serves as the endpoints of some tunnels, it has to deal with the aggregated traffic passing through the tunnels rather than just the traffic sent to and received from a single host. Worse yet, cryptographic operations are usually computation-intensive and are likely to consume a large amount of processing power even if only a small volume of traffic is being processed [4][5][6].

The drawback of poor scalability did not cause significant problems until recently in that the performance of a VPN was generally limited by the slowest link, which was one of the last-mile connections. Under such circumstances, security gateways implemented in low-cost RISC architectures with software encryption/decryption were adequate to the job. However, as broadband services like Fiber To The Home/Fiber To The Building (FTTH/FTTB) [7] become more and more popular, the bandwidth between a network and Internet backbone may grow beyond tens or even hundreds of Mbps in the near future. Since the original bottleneck is removed, the security gateways now become the new bottleneck.

In this paper, we propose a clustering scheme to scale the throughput of CPE-based IPsec VPNs. The proposed scheme aggregates the processing power of multiple security gateways by means of a transparent self-dispatching technique and hence allows as many gateways to be added as necessary until the resulting throughput is again limited by the bandwidth of the last-mile connections. We also create our own flow-migration mechanism to keep the load of the gateways balanced. The rest of the paper is organized as follows. Section 2 gives a brief introduction to some important research efforts in this area, especially

those devoted to improving the performance of security gateways. Section 3 describes various parts of the proposed scheme in great details, followed by the performance evaluation presented in Section 4. Section 5 then concludes the paper by summarizing our achievements.

2 Related Work

For the purpose of improving VPN scalability, researchers and vendors have created various techniques and different provisioning models. In this section, we first introduce a relatively new type of VPNs, the network-based IP VPNs, which gains much market interest, and then discuss hardware-assisted acceleration technologies for IPsec processing.

2.1 Network-Based IP VPNs

In order to prevent customers with high-volume VPN traffic from paying for expensive security gateways, service providers and equipment vendors proposed the idea of network-based IP VPNs [8][9]. Instead of requiring each customer to acquire a separate security gateway, service providers deploy powerful provider-edge VPN devices that are capable of sustaining a large number of concurrent tunnels and begin to promote secure data transmission as a new service [10]. Thus, VPN tunnels terminate on provider-edge devices and only regular routers are needed on the customer edges. Therefore, customers may get rid of the cost incurred by the deployment and maintenance of specialized security gateways.

The major issue of network-based IP VPNs is the tradeoff between security and scalability [11]. When customers delegate the responsibility for deploying, configuring, and maintaining the devices that are endpoints of VPN tunnels, it is implied that customers must trust their service providers because data gets encrypted after it enters the networks of service providers and service providers may have a chance to inspect or modify the data to be transferred. (In fact, service providers will certainly alter the data in order to transform it into the encrypted form.) Unfortunately, service providers are not always trustworthy, and there are also customers who want better control over the data entering/leaving their networks. After all, security is the primary reason why customers are willing to pay for VPNs in the first place. The proposed scheme does not make such tradeoff. That is, it improves the scalability of CPE-based IPsec VPNs without sacrificing the security.

2.2 Hardware-Accelerated IPsec Processing

Software implementation of packet processing functions is often criticized for their low performance. Therefore, as the IPsec protocol becomes mature, it is natural for researchers and vendors to invent new hardware that accelerates the cryptographic primitives used in IPsec processing. Such performance-enhancing

approach is often classified as the scale-up approach [12] because the primary focus is to increase the computation power of a single unit.

Various designs of IPsec-acceleration ASICs and architectures have been proposed and evaluated. [13] proposed a cryptographic coprocessor specialized for AES processing with a maximum throughput of 3.43 Gbps, and [14] described the design of a hardware accelerator which supported many algorithms specified in the IPsec standard, including AES, 3DES, HMAC-MD5 and HMAC-SHA1. [15] compared different hardware architectures for high-performance VPN devices and estimated the total cost of each. [16] presented the benchmarking results of a security-gateway implementation over the IXP425 network processor developed by Intel Corporation, and [17] evaluated the performance of several commercial products, some of which were also implemented on the IXP425 network processors. [18] discussed another security-gateway implementation using Intel IXP1200 network processor, a more powerful one in the family.

Although specialized hardware can be created to improve the performance of certain cryptographic primitive to a great extent, the ASIC-design approaches are also infamous for their inflexibility. If the protocol evolves or a flaw in the design is found after the production of certain ASIC, the revision of the ASIC will have to restart an entire silicon spin, which is both time-consuming and costly. Alternatively, the emerging technology of network processors [19] attempts to overcome the limitation of inflexibility by introducing highly programmable hardware and seems to have successfully drawn much attention. However, as the flexibility increases, the complexity of programming also raises. Besides, vendors have created a wide variety of incompatible programming models, resulting in long learning curves of system integration even for experienced professionals.

In contrast to the scale-up approach, our scheme follows the scale-out approach [12], in which multiple units cooperate to allow higher throughput of IPsec processing, and does not suffer the limitations mentioned above. In addition, it does not interfere with hardware acceleration and hence can be combined with the scale-up approach to achieve even higher performance.

3 The Proposed Scheme

In this section, we start by introducing the reference system architecture and then discuss two important aspects, the techniques for transparently dispatching outbound VPN traffic and the mechanisms for traffic redistribution. Subsequently, we describe the proposed scheme in detail.

3.1 System Architecture

Fig. 2 shows a part of a CPE-based VPN from the perspective of the site that interests us. It corresponds to the left half of Fig. 1 with the difference that the single security gateway on which VPN tunnels terminate is replaced by a cluster of multiple security gateways. The cluster is constructed by surrounding all member gateways with two Ethernet switches and adding a third network

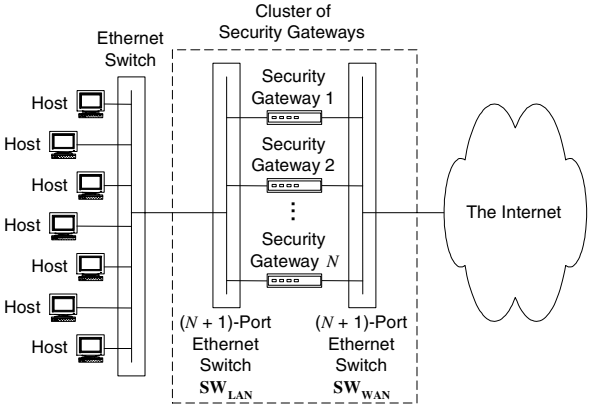

Fig. 2. The cluster is constructed by surrounding multiple security gateways with switches so that it can replace a single security gateway without changing the network topology external to the cluster

interface to each gateway for interconnection. (The third interfaces, the switch used for interconnection, and the cables are not explicitly shown in Fig. 2.) The construction of the cluster is so designed that it can be inserted into the exact location in which the original security gateway resides.

As shown in Fig. 2, there are N security gateways in the cluster. We assume that there are also N security gateways in the remote cluster (i.e., the other side of the VPN). Furthermore, for each tunnel terminating on the original gateways in Fig. 1, there is a counterpart between each pair of the ith gateway in the local cluster and the ith gateway in the remote cluster, where $1 \leq i \leq N$. That is, outbound VPN traffic sent from the ith gateway in one cluster is always destined for the ith gateway in the other cluster. The purpose of making these assumptions is merely to keep the following description away from irrelevant complications. The assumptions can be easily relaxed with minor adjustments.

3.2 Traffic-Dispatching Techniques

In a typical site-to-site scenario of CPE-based IPsec VPNs, when a host has a packet to send to the other side of the VPN, it first looks up $\mathbf{IP_G}$, the IP address of the security gateway, in its routing table. Then the host determines $\mathbf{MAC_G}$, the layer-2 MAC address corresponding to $\mathbf{IP_G}$, by excercising the ARP protocol. After the host gets $\mathbf{MAC_G}$, it encapsulates the packet with a properly constructed frame and places the frame on the wire.

Since each host generally follows the procedure described above, two techniques can be used to distribute the traffic generated by a group of hosts over multiple security gateways. First, we may assign different $\mathbf{IP_G}$'s to individual hosts by manipulating their routing-table entries so that the frames sent from the hosts will be destined for different gateways by default. Second, we may implement customized ARP processing modules on the security gateways so that

individual hosts will receive different values of $\mathbf{MAC_G}$ for the same $\mathbf{IP_G}$. Note that both techniques make the frames self-dispatched and eliminate the need for centralized dispatchers.

A major disadvantage of the first technique is the lack of transparency. This actually defeats the purpose of making a cluster. On the contrary, when the second technique is adopted, the clustered security gateways share the same IP address, $\mathbf{IP_G}$, on the LAN side (i.e. on the network interfaces connected to $\mathbf{SW_{LAN}}$, which is the left switch within the cluster in Fig. 2) and present the consistent image of a single but more powerful gateway to the hosts. the second technique also limits any customization within the cluster and hence does not require the hosts to change their original behavior. In addition, since most hosts implement ARP processing in a soft-state fashion, the second technique will have less trouble when the bindings between $\mathbf{IP_G}$ to various $\mathbf{MAC_G}$'s need to be altered. Therefore, the proposed scheme adopts the second traffic-dispatching technique. Section 4.2 demonstrates the effectiveness of this choice.

The two traffic-dispatching techniques mentioned above have their corresponding traffic-redistribution mechanisms. When the first technique is used, it is possible for the security gateways to indirectly update the routing tables of the hosts via ICMP Redirect messages from time to time. However, such traffic-redistribution mechanism requires cooperation from the hosts and is less feasible. As for the second traffic-dispatching technique, we can change the behavior of the customized ARP processing modules on the security gateways to always replying with the MAC address of the least-loaded gateway when responding to ARP requests. The major drawback of this mechanism is its slow response to the variation of the traffic pattern in that it acts passively and the effectiveness depends on the timeout value of the ARP-cache entries stored in individual hosts. Since both mechanisms are not good enough, we decide to create a third mechanism, which is described as follows.

3.3 Clustering with Traffic Redistribution

As shown in Fig. 2, the cluster of the security gateways also plays the role of the edge router. Therefore, in the phase of initial setup, the Ethernet interfaces connected to $\mathbf{SW_{LAN}}$ are configured to share $\mathbf{IP_G}$ as their common IP address, and those interfaces connected to $\mathbf{SW_{WAN}}$ are configured to have distinct public IP addresses. The routing table and the VPN software of each gateway are configured to set up VPN tunnels in the way mentioned in Section 3.1. Once properly configured, each of the security gateways is able to process VPN traffic to and from the tunnels and also route non-VPN traffic to and from the Internet. Some final steps of the initial setup are depicted in the following algorithm.

Algorithm 1. Additional preparation work in the phase of initial setup

1. Generate $\mathbf{V}$, a set of K virtual MAC addresses, where K is a sufficiently large number, compared with the number of Ethernet interfaces in the LAN in which the cluster resides. Each address must be unique in the same segment. (Locally administered Ethernet addresses are good choices.)

2. Choose h, a hash function that takes m, a MAC address, as the input and maps it to v, an element in $\mathbf{V}$. That is,

$$h: \ \mathbf{M} \ \rightarrow \ \mathbf{V}, \quad \text{where } \mathbf{M} = \{m \mid m \in \mathbb{Z}, \ 0 \leq m \leq 2^{48} - 1\}.$$

3. Equally partition $\mathbf{V}$ into subsets $\mathbf{V}_1, \mathbf{V}_2, \mathbf{V}_3, \cdots, \mathbf{V}_N$, where N is the number of security gateways in the cluster.

Succeeding the phase of initial setup, each security gateways in the cluster runs the following four algorithms.

Algorithm 2. Frame processing

1. Define a flow, noted as $\mathbf{F}_m$, to be the set of all Ethernet frames sharing a common destination MAC address, m.
2. **In the outbound direction:** For each element v in $\mathbf{V}_i$, modify the frame-processing module running on $\mathbf{G}_i$, the ith security gateway, $1 \leq i \leq N$, to additionally accept and process the Ethernet frames (i.e., to pass the received frames to upper-layer VPN software) belonging to $\mathbf{F}_v$. $\mathbf{G}_i$ is referred to as the responsible owner of $\mathbf{F}_v$.
3. **In the inbound direction:** Modify the frame-processing module running on $\mathbf{G}_i$ in such a way that every frame sent to a destination MAC address m in the LAN will always have $h(m)$ as its source MAC address.

Algorithm 3. Port-learning trigger

1. For each element v in $\mathbf{V}_i$, set up a timer routine on $\mathbf{G}_i$ to periodically transmit unsolicited Ethernet frames with source MAC address being v to refresh the forwarding table of $\mathbf{SW_{LAN}}$ so that flooding can be avoided.

Algorithm 4. ARP Processing

1. Upon receiving an ARP request with respect to $\mathbf{IP_G}$, each $\mathbf{G}_i$ extracts the source MAC address m from the request and computes $x = h(m)$.
2. If $x \in \mathbf{V}_i$ (i.e., $\mathbf{G}_i$ is the responsible owner of $\mathbf{F}_x$), $\mathbf{G}_i$ sends back an ARP reply with the answer being x. Otherwise, $\mathbf{G}_i$ ignores the ARP request.

The above three algorithms together create the illusion of a super gateway device with many built-in Ethernet interfaces. With carefully selected K and h, each host in the LAN can be viewed as being directly connected to an individual Ethernet interface of the emulated super gateway device. These algorithms take advantage of the self-learning procedure implemented in every transparent bridge [20][21] and successfully realize fully decentralized traffic distribution. The details of traffic redistribution are described as follows.

Algorithm 5. Flow migration

1. Define the predecessor of $\mathbf{G}_i$, which is denoted by $p(\mathbf{G}_i)$, to be $\mathbf{G}_{i-1}$ when $2 \leq i \leq N$ and $\mathbf{G}_N$ when $i = 1$. Define the successor of $\mathbf{G}_i$, which is denoted by $s(\mathbf{G}_i)$, to be $\mathbf{G}_{i+1}$ when $1 \leq i \leq N - 1$ and $\mathbf{G}_1$ when $i = N$.

2. For every T_L seconds, $\mathbf{G}_i$ sends its load information to $p(\mathbf{G}_i)$ through the Ethernet interface designated for inter-gateway communication.
3. For every T_M seconds, $\mathbf{G}_i$ makes a flow-migration decision based on the load information it has. If the load of $\mathbf{G}_i$ is larger than the load of $s(\mathbf{G}_i)$ and the difference is greater than a threshold value L_D, then $\mathbf{G}_i$ randomly generates a subset $\mathbf{W}$ (with a fixed size) of $\mathbf{V}_i$ and notifies $s(\mathbf{G}_i)$ about its selection through the Ethernet interface designated for inter-gateway communication. Otherwise, $\mathbf{G}_i$ does nothing and waits until the next round.
4. Upon receiving a notification message from $p(\mathbf{G}_i)$, $\mathbf{G}_i$ extracts $\mathbf{W}$ from the message and, for each element w in $\mathbf{W}$, $\mathbf{G}_i$ inserts it into $\mathbf{V}_i$ (i.e., $\mathbf{G}_i$ becomes the new responsible owner for $\mathbf{F}_w$) and also sends an acknowledging Ethernet frame (with source MAC address being w) back to $p(\mathbf{G}_i)$ via $\mathbf{SW_{LAN}}$.
5. Upon receiving an acknowledging frame from $s(\mathbf{G}_i)$, $\mathbf{G}_i$ extracts the source MAC address w, which is actually one of the elements previously picked by $\mathbf{G}_i$ itself, from the frame and remove w from $\mathbf{V}_i$.

As depicted in Algorithm 5, traffic redistribution is accomplished by reassigning the flows to new responsible owners. The acknowledging frame sent in step 4 serves dual purposes. In addition to informing $p(\mathbf{G}_i)$ that $\mathbf{G}_i$ has completed the processing of the flow migration regarding w, the acknowledging frame also updates the forwarding table of $\mathbf{SW_{LAN}}$ so that $\mathbf{SW_{LAN}}$ will forward later frames destined for w through the port to which $\mathbf{G}_i$ is connected instead of the original port to which $p(\mathbf{G}_i)$ is connected. Again, Algorithm 5 relies on the self-learning behavior of $\mathbf{SW_{LAN}}$.

4 Performance Evaluation

We implement the proposed scheme as collaborating kernel modules running with Linux kernel (version 2.4.29). Openswan (version 1.0.9) is chosen to be the VPN software. The environment and the results of tests are presented as follows.

4.1 Test Environment

Fig. 3 shows the network diagram of the test environment. We use the world-recognized test equipment, SmartBits, from Spirent Communications, Inc., to generate a large volume of traffic in each test. In the lower half of Fig. 3, each unit is an x86 machine equipped with an Intel Xeon 2.4 GHz processor. The three machines on the left and the three machines on the right comprise two clusters, respectively, and the machine in the middle is configured as a router that simulates the Internet backbone. The PC in Fig. 3 runs the SmartBits control application, SmartFlow (version 3.0), on top of Microsoft Windows XP.

4.2 The Effectiveness of Clustering

In this test, we use SmartBits to simulate 224 individual hosts on each side of the VPN. Every simulated host has a distinct IP address and a unique MAC

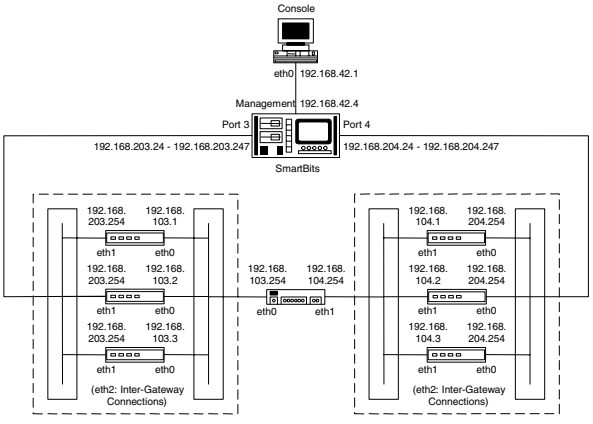

Fig. 3. SmartBits is used to generate a large volume of traffic that passes through the clusters of the security gateways in both directions

address, and will keep sending packets to its counterpart residing in the other side of the VPN once the test starts. The IP addresses and the MAC addresses of the simulated hosts on the left side of the VPN occupy the continuous ranges of 192.168.203.24–247 and 00:00:03:00:00:18–F7, respectively, and the IP addresses and the MAC addresses of the simulated hosts on the right side of the VPN occupy the continuous ranges of 192.168.204.24–247 and 00:00:04:00:00:18–F7, respectively. The endpoint pairs of the tunnels are (192.168.103.1, 192.168.104.1), (192.168.103.2, 192.168.104.2), and (192.168.103.3, 192.168.104.3), respectively.

Since the traffic pattern is fixed over time, we temporarily disable the traffic-redistribution functionality (i.e., Algorithm 5) and focus on the scalability of the chosen traffic-dispatching technique. We set $K = 256$ and make $\mathbf{V}$ consist of all MAC addresses in the range of 02:88:88:88:88:00–FF. We also choose h to be

$$h(m) = \text{0x028888888800} \mid (m \ \% \ K), \quad \begin{array}{l}\text{where } \mid \text{ is the bitwise-or operator and} \\ \% \text{ is the modulo operator.}\end{array}$$

Then we measure the zero-loss throughput under the combinations of three different cluster sizes, two different frame sizes, and two popular types of IPsec ESP tunnels. Fig. 4 shows the results.

The numbers and the bars in Fig. 4 show the effectiveness of the chosen traffic-dispatching scheme. The proposed scheme consistently exhibits high scalability under all combinations. Taking the first group, 3DES-SHA1 tunnels with input frame size being 64 bytes, as an example. When the cluster size increases from 1 to 2, the zero-loss throughput is doubled. As we put three gateways into each cluster, the zero-loss throughput becomes 2.98 times, almost tripled.

4.3 The Effectiveness of Traffic Redistribution

Two parameters used in this test are different from those used in the previous test. First, the traffic-redistribution functionality is enabled. Second, the MAC

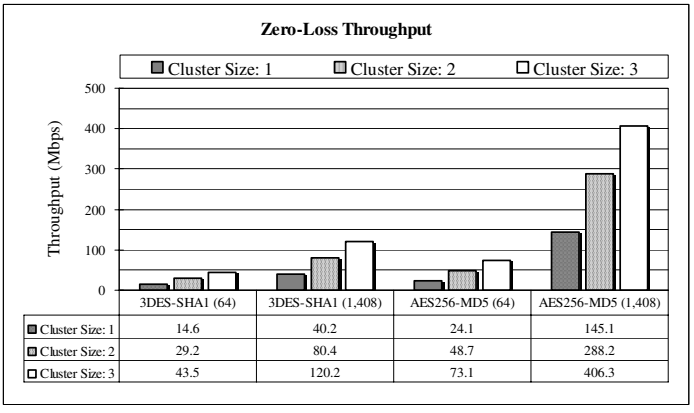

Fig. 4. Zero-loss throughput of the proposed clustering scheme with static traffic-distribution policy grows linearly as the cluster size increases

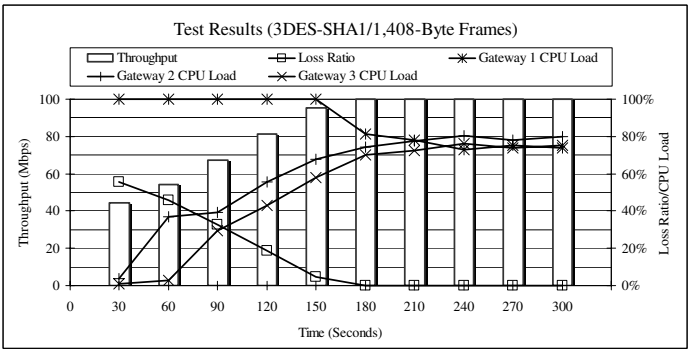

Fig. 5. Test results show that the overall throughput keeps growing to its maximum as the traffic is getting more and more evenly distributed

addresses of the simulated hosts on the two sides of the VPN are now the two arithmetic sequences 00:00:03:00:18:00, 00:00:03:00:19:00, 00:00:03:00:1A:00, $\cdots$, 00:00:03:00:F7:00 and 00:00:04:00:18:00, 00:00:04:00:19:00, 00:00:04:00:1A:00, $\cdots$, 00:00:04:00:F7:00, respectively. Using these addresses with the chosen h makes the first security gateway in each cluster become the initial responsible owner of all flows when the test starts.

This time we use 3DES-SHA1 tunnels, a value of 3 for N, a value of 1.33 for T_L, a value of 4 for T_M, a value of 10% for L_D, and a value of 4 for the size of $\mathbf{W}$. We also take a snapshot every 30 seconds during the test. When the test starts, we instruct SmartBits to generate bidirectional 100 Mbps of traffic in the form of 1,408-byte frames for 300 seconds. The test results are shown in Fig. 5.

In Fig. 5, the bars designate the measured throughput of the bidirectional VPN traffic in the snapshots taken every 30 seconds. The values of the bars

can be found on the left vertical axis in the figure. For example, the value of the rightmost bar in Fig. 5 is 100 Mbps. That means the two clusters forming the VPN are able to process 100 Mbps of left-to-right VPN traffic and another 100 Mbps of right-to-left VPN traffic at the same time. The four lines in the figure represent the ratio of frame loss and the CPU load of individual gateways in the left cluster, respectively. The values of the data points in the lines can be found on the right vertical axis in the figure.

The bars show a climbing trend of overall throughput and, around the 180th second, the throughput reaches its maximum of 100 Mbps, which is exactly the same as the offered traffic. The lines representing the CPU load provide an evident trace of the progression of traffic redistribution. As soon as a sufficient number of flows have been reassigned and the CPU load of the first gateway drops below 100%, there becomes no frame loss at all.

5 Conclusions

In this paper, we propose a scale-out scheme for constructing high-performance CPE-based IPsec VPNs via clustering and traffic distribution. The demand for higher-throughput security-gateway clusters arises because of two factors. First, as the bandwidth of WAN links keeps growing rapidly, the last-mile connections are now no longer the bottleneck. After the original bottleneck is removed, the edge devices in IPsec VPNs become the new bottleneck. Second, scale-up approaches will quickly hit various physical, electrical, and electronic limitations, and have the disadvantages of inflexibility as well as high software complexity.

The proposed scheme groups a number of security gateways together to form a clustering solution so that each cluster is capable of processing a large volume of VPN traffic. High throughput is the result of both aggregating the processing capability of multiple gateways and redistributing the outbound VPN traffic when the load of the gateways becomes unbalanced. Aggregation of the processing power is realized by a transparent self-dispatching technique that requires no dispatchers, and traffic redistribution is accomplished by flow migration. The proposed scheme is adaptive, scalable, and transparent. It responds to the variation of the traffic pattern promptly, and the throughput grows linearly as the number of security gateways in the cluster increases. All customization is limited within the clusters and no change to any other components outside the clusters is needed. In addition, the proposed scheme does not suffer the limitations of the scale-up approaches and can be combined with hardware-accelerated solutions to achieve even higher performance.

References

1. Ortiz, Jr., S.: Virtual private networks: Leveraging the Internet. IEEE Computer **30** (1997) 18–20
2. Kent, S., Atkinson, R.: Security architecture for the Internet protocol. RFC 2401 (1998)

3. Knight, P., Lewis, C.: Layer 2 and 3 virtual private networks: Taxonomy, technology, and standardization efforts. IEEE Communications Magazine **42** (2004) 124–131
4. Elkeelany, O., Matalgah, M.M., Sheikh, K.P., Thaker, M., Chaudhry, G., Medhi, D., Qaddour, J.: Performance analysis of IPSec protocol: Encryption and authentication. In: Proceedings of 2002 IEEE International Conference on Communications (ICC 2002). Volume 2. (2002) 1164–1168
5. Lin, J.C., Chang, C.T., Chung, W.T.: Design, implementation and performance evaluation of IP-VPN. In: Proceedings of 17th International Conference on Advanced Information Networking and Applications (AINA 2003). (2003) 206–209
6. Khanvilkar, S., Khokhar, A.: Virtual private networks: An overview with performance evaluation. IEEE Communications Magazine **42** (2004) 146–154
7. Kettler, D., Kafka, H., Spears, D.: Driving fiber to the home. IEEE Communications Magazine **38** (2000) 106–110
8. Metz, C.: The latest in virtual private networks: Part I. IEEE Internet Computing **7** (2003) 87–91
9. Metz, C.: The latest in virtual private networks: Part II. IEEE Internet Computing **8** (2003) 60–65
10. Carugi, M., De Clercq, J.: Virtual private network services: Scenarios, requirements and architectural constructs from a standardization perspective. IEEE Communications Magazine **42** (2004) 116–122
11. De Clercq, J., Paridaens, O.: Scalability implications of virtual private networks. IEEE Communications Magazine **40** (2002) 151–157
12. Devlin, B., Gray, J., Laing, B., Spix, G.: Scalability terminology: Farms, clones, partitions, and packs: RACS and RAPS. Technical Report MS-TR-99-85, Microsoft Research (1999)
13. Hodjat, A., Verbauwhede, I.: High-throughput programmable cryptocoprocessor. IEEE Micro **24** (2004) 34–45
14. Ha, C.S., Lee, J.H., Leem, D.S., Park, M.S., Choi, B.Y.: ASIC design of IPSec hardware accelerator for network security. In: Proceedings of 2004 IEEE Asia-Pacific Conference on Advanced System Integrated Circuits (AP-ASIC 2004). (2004) 168–171
15. Friend, R.: Making the gigabit IPsec VPN architecture secure. IEEE Computer **37** (2004) 54–60
16. Lin, Y.N., Lin, C.H., Lin, Y.D., Lai, Y.C.: VPN gateways over network processors: Implementation and evaluation. In: Proceedings of 11th IEEE Real-Time and Embedded Technology and Applications Symposium (RTAS 2005). (2005) 480–486
17. The Tolly Group, Inc.: Intel IXP425 network processors: Performance analysis of VPN devices. Document No. 204132 (2004)
18. Han, M., Kim, J., Sohn, S.: Network processor for IPSec. In: Proceedings of 6th International Conference on Advanced Communication Technology (ICACT 2004). Volume 1. (2004) 485–487
19. Comer, D.E.: Network Systems Design Using Network Processors. Pearson Prentice Hall, Inc. (2003)
20. IEEE Standards Association: IEEE standard for local and metropolitan area networks: Media access control (MAC) bridges. IEEE 802.1D-2004 (2004)
21. Seifert, R.: The Switch Book: The Complete Guide to LAN Switching Technology. John Wiley & Sons, Inc. (2000)